This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims

1. (Previously Presented) A method for interrupt processing, comprising:

determining that an event has occurred;

determining a processor identifier;

determining an event data structure identifier for an event data structure into which data for the event is stored using the processor identifier, wherein the event data structure identifier is determined by accessing a message vector mapping structure using the processor identifier and an event code;

determining a vector identifier for an interrupt message vector; and

writing interrupt message data to the interrupt message vector to generate an interrupt.

2. (Original) The method of claim 1, wherein the processor identifier is determined by applying a hash technique to a data packet to access a processor redirection/indirection structure.

3. (Cancelled)

4. (Original) The method of claim 1, wherein the vector identifier is determined by accessing a message vector mapping structure using the processor identifier and an event code.

5. (Previously Presented) The method of claim 6, wherein the event data structure identifier is determined by accessing a processor redirection/indirection structure using the processor identifier.

6. (Previously Presented) A method for interrupt processing, comprising:

determining that an event has occurred;

determining a processor identifier;

determining an event data structure identifier for an event data structure into which data for the event is stored using the processor identifier;

determining a vector identifier for an interrupt message vector, wherein the vector identifier is determined from a message vector mapping structure using the event data structure identifier as an index; and

writing interrupt message data to the interrupt message vector to generate an interrupt.

7. (Original) The method of claim 1, further comprising:

writing an event entry to the event data structure identified by the event data structure identifier; and

advancing a write indicator.

8. (Original) The method of claim 1, further comprising:

receiving an interrupt;

identifying an event data structure using the interrupt message data in the interrupt message vector; and

processing an event entry in the identified event data structure.

9. (Original) The method of claim 1, further comprising:

determining whether the event is associated with data; and

determining a default processor identifier in response to determining that the event is not associated with data.

10. (Currently Amended) A system for [[in]] interrupt processing, comprising:

an Input/Output device coupled to a Network Interface Controller bus; and

circuitry at the Input/Output device operable to:

determine that an event has occurred;

determine a processor identifier from a processor redirection/indirection structure;

determine an event data structure identifier for an event data structure into which data for the event is stored using the processor identifier, wherein the event data structure identifier is determined by accessing a message vector mapping structure using the processor identifier and an event code;

determine a vector identifier for an interrupt message vector into which an interrupt message is written; and

write interrupt message data to the interrupt message vector to generate an interrupt.

11. (Previously Presented) The system of claim 10, wherein the processor identifier is determined by applying a hash technique to a data packet to access a processor redirection/indirection structure.

12. (Cancelled)

13. (Previously Presented) The system of claim 10, wherein the vector identifier is determined by accessing a message vector mapping structure using the processor identifier and an event code.

14. (Previously Presented) The system of claim 15, wherein the event data structure identifier is determined by accessing a processor redirection/indirection structure using the processor identifier.

15. (Currently Amended) A system for [[in]] interrupt processing, comprising:

an Input/Output device coupled to a <u>Network Interface Controller</u> bus; and

circuitry at the Input/Output device operable to:

determine that an event has occurred;

determine a processor identifier from a processor redirection/indirection structure;

determine an event data structure identifier for an event data structure into which data for the event is stored using the processor identifier;

determine a vector identifier for an interrupt message vector into which an interrupt message is written, wherein the vector identifier is determined from a message vector mapping structure using the event data structure identifier as an index; and write interrupt message data to the interrupt message vector to generate an interrupt.

16. (Previously Presented) The system of claim 10, wherein the circuitry is operable to:

write an event entry to the event data structure identified by the event data structure identifier; and

advance a write indicator.

17. (Previously Presented) The system of claim 10, further comprising:

an Input/Output device driver coupled to a bus; and

circuitry at the Input/Output device driver operable to:

receive an interrupt;

identify an event data structure using the interrupt message data in the interrupt message vector; and

process an event entry in the identified event data structure.

18. (Previously Presented) The system of claim 10, wherein the circuitry is operable to:

determine whether the event is associated with data; and

determine a default processor identifier in response to determining that the event is not associated with data.

19. (Previously Presented) An article of manufacture for interrupt processing, wherein the article of manufacture is at an Input/Output device and is operable to:

determine that an event has occurred;

determine a processor identifier from a processor redirection/indirection structure;

determine an event data structure identifier for an event data structure into which data for the event is stored using the processor identifier, wherein the event data structure identifier is determined by accessing a message vector mapping structure using the processor identifier and an event code;

determine a vector identifier for an interrupt message vector into which an interrupt message for the event is stored; and

write interrupt message data to the interrupt message vector to generate an interrupt .

20. (Original) The article of manufacture of claim 19, wherein the processor identifier is determined by applying a hash technique to a data packet to access a processor redirection/indirection structure.

21. (Cancelled)

22. (Original) The article of manufacture of claim 19, wherein the vector identifier is determined by accessing a message vector mapping structure using the processor identifier and an event code.

23. (Previously Presented) The article of manufacture of claim 24, wherein the event data structure identifier is determined by accessing processor redirection/indirection structure using the processor identifier.

24. (Previously Presented) An article of manufacture for interrupt processing, wherein the article of manufacture is at an Input/Output device and is operable to:

determine that an event has occurred;

determine a processor identifier from a processor redirection/indirection structure;

determine an event data structure identifier for an event data structure into which data for the event is stored using the processor identifier;

determine a vector identifier for an interrupt message vector into which an interrupt message for the event is stored, wherein the vector identifier is determined from a message vector mapping structure using the event data structure identifier as an index; and

write interrupt message data to the interrupt message vector to generate an interrupt.

25. (Original) The article of manufacture of claim 19, wherein the article of manufacture is operable to:

write an event entry to the event data structure identified by the event data structure identifier; and

advance a write indicator.

26. (Original) The article of manufacture of claim 19, wherein the Input/Output device is connected to a device driver and wherein an article of manufacture at the Input/Output device driver is operable to:

receive an interrupt;

identify an event data structure using the interrupt message data in the interrupt message vector; and

process an event entry in the identified event data structure.

27. (Original) The article of manufacture of claim 26, wherein the article of manufacture is operable to:

determine whether the event is associated with data; and

determine a default processor identifier in response to determining that the event is not associated with data.